Towards Context-Aware Data Management for Ambient Intelligence

Ling Feng, Peter M.G. Apers, and Willem Jonker

Dept. of Computer Science, University of Twente
PO Box 217, 7500 AE Enschede, The Netherlands
{ling,apers,jonker}@cs.utwente.nl

Abstract. *Ambient Intelligence* (AmI) is a vision of future Information Society, where people are surrounded by an electronic environment which is sensitive to their needs, personalized to their requirements, anticipatory of their behavior, and responsive to their presence. It emphasizes on greater user-friendliness, user-empowerment, and more effective service support, with an aim to make people's daily activities more convenient, thus improving the quality of human life. To make AmI real, effective data management support is indispensable. High-quality information must be available to any user, anytime, anywhere, and on any lightweight device. Beyond that, AmI also raises many new challenges related to context-awareness and natural user interaction, entailing us to re-think current database techniques.
The aim of this paper is to address the impact of AmI, particularly its *user-centric context-awareness* requirement on data management strategies and solutions. We first provide a multidimensional view of database access context. Taking diverse contextual information into account, we then present five context-aware data management strategies, using the most fundamental database operation - context-aware query request as a case in point. We execute the proposed strategies via a two-layered infrastructure, consisting of *public data manager*(s) and a *private data manager*. Detailed steps of processing a context-aware query are also described in the paper.

1 Introduction

Developments in *ubiquitous computing* and *ubiquitous communication*, together with *intelligent user friendly interfaces*, will eventually lead to a world in which computing functionality will be embedded in all kinds of objects, which are capable of recognizing and responding to individual human needs in a seamless, unobtrusive and often invisible way. An example is a hotel room that can adapt automatically to its customer's favorite room temperature and music choice. Such a vision of the future was coined **Ambient Intelligence** (AmI) by the European Community's Information Society Technology (IST) Program, whose aim is to sustain and extend the objectives of the eEurope 2002 Action Plan of

F. Galindo et al. (Eds.): DEXA 2004, LNCS 3180, pp. 422–431, 2004.

bringing IST applications and services to everyone, every home, every school, and every business [4].

The aim of this paper is to address the impact of AmI, particularly its *user-centric context-awareness* requirement on data management strategies and solutions. Unlike conventional data management paradigms, ambient data management frees users from the constraint of stationary desktops, enabling data sharing and dissemination throughout the ambience of users. The database access in an AmI environment will not occur at a single location in a single context, as in desktop computing, but rather span a multitude of contexts covering the office, plane, meeting room, home, hotel, and so on.

Till now, decades of efforts have been made in improving *content*-based database access, due to the long-historical stationary database constraint. Nevertheless, AmI promotes us to go further for *context*-based data access. "*Get the report which I prepared last night before dinner in the hotel for this afternoon's meeting*" and "*Find restaurants nearby which I have not visited for half a year*" are two examples of such queries. Identifying data items by context has great potential, especially for the coming AmI era. On the one hand, it facilitates users to make the best of data through a flexible query-answering mechanism. On the other hand, it provides hints on how to process data requests in the most optimal way, as it carries a kind of semantics related to what, why, when, where, and how to use data sources. We believe that by context awareness, the interaction between data managers and users can be enriched than ever.

To deliver context-aware data management solutions raises a number of interesting and challenging questions as follows.
a) By context-awareness, can we make data managers more adaptable, responsive, personalized, dynamic, and anticipatory, as charted by AmI, than before?
b) Compared with traditional data management, what are the fundamental issues underlying context-aware data management?
c) To bring context-awareness feature into data management, how to acquire, categorize, and model contextual information?
d) How to exploit contexts to answer a user's data request?
e) What are context-aware data management strategies? How to support, manage, and execute these strategies?
f) How to provide context-aware data management supports to users? How to design a friendly and easy-to-use context-aware query language for users? How to effectively and efficiently communicate with users, given a handheld device which is much smaller than conventional desktop devices?

The purpose of this study is to propose potential ways to tackle the above problems. In Section 2, we review some closely related work. A multidimensional view of context is described in Section 3. Taking the diverse contexts into account, five context-aware data management strategies are proposed in Section 4. To execute these strategies, a two-layered infrastructure is given in Section 5. We overview different steps of processing a context-aware query in Section 6. We conclude the paper in Section 7.

2 Related Work

Context awareness is being actively studied in different fields. In this section, we review related work from a data management perspective. More comprehensive good survey on context-aware computing can be found in [3,20].

Location-Dependent Query Processing. Location is an important kind of context. The concept of location constrained queries, i.e., queries whose answers depend on the locations of mobile users, was first introduced in [16]. [7,26] presented a location-dependent query processing architecture. An approach of querying the nearest services in a multi-cell mobile environment was given in [32]. [27] designed a Moving Objects Spatial-Temporal data model for querying moving objects. [28,17] investigated dynamic queries over mobile objects, where the locations of query issuers themselves are changing with the time.

Conceptual Modeling of Context Information. [23] proposed to model contextual information using *key-value* pairs. [13] structured contextual information based on entity-relationship paradigm. An object-oriented modeling technique was explored to model networked environments [12]. [11] represented sensed context using first-order predicate logic, composed into more complex sensed context expressions and associated with meta-propositional properties.

Context-Aware Computing Frameworks. Early attempts included Active Badge Infrastructure [24] and Stick-e Notes framework [2]. The *widget, server,* and *interpreter* components in the Context Toolkit separated context acquisition from the delivery and use of context [6]. A four-layered model, consisting of *sensors, cues, context profiles,* and *an application layer*, was further proposed [25]. [15] developed a Situated Computing Service to encapsulate context acquisition from applications. A middleware approach [14], as well as an agent-based architecture [21], was also presented for context-aware computing.

Context-Aware Applications. Most currently existing applications use identity, location, and time. The sensors used are mainly short range IR and RF signals, and GPS. These applications include Active Badge [30], office assistant ParcTab [31], shopping guide [1], In/Out registration board [22], Conference Assistant [6], tour-guide [6,5], the reminder and memory aid system [10,19].

3 Context Categorization, Acquisition, and Modeling

In this study, we are aiming at context-aware database support for AmI. The term *context* here refers to the situation under which user's database access happens. We categorize context into two kinds, namely, *user-centric context* and *environmental context*, as depicted in Figure 1. We view context as an n-dimensional space, constructed by n contextual attributes. Each dimension of the context is represented by one contextual attribute, describing one perspective of context. The domain of a contextual attribute can be either a scalar value, a string value, a set of scalar/string values, or an empty or a NULL value, depending on the application semantics. For example, the domain of the contextual attribute *trafficJam* can be a set of strings, specifying the names of sites where a traffic jam happens. When there is no traffic jam, *trafficJam*=$\emptyset$.

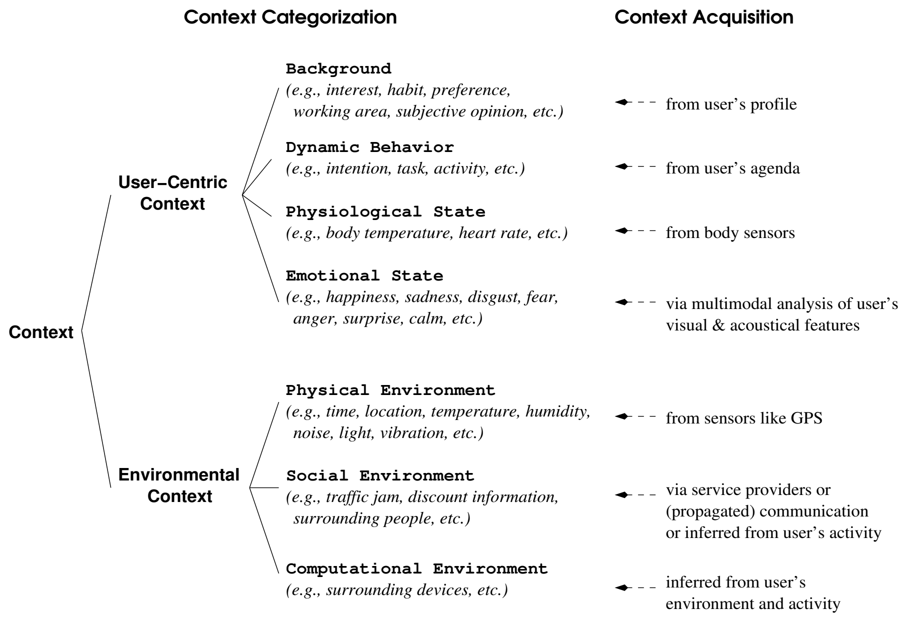

Fig. 1. Context categorization and acquisition

Formally, let $\{a_1, a_2, \ldots, a_n\}$ be a set of contextual attributes, whose domains are represented as $Dom(a_1)$, $Dom(a_2)$, $\ldots$, $Dom(a_n)$. An *n-dimensional context*, denoted as $[\![\mathcal{C}ontext]\!]$, is the cartesian product $[\![\mathcal{C}ontext]\!] : a_1 \times a_2 \times \ldots \times a_n$ of these contextual attributes. According to application requirements, abstract operators can be defined on compatible contextual attributes. To support operations conducted on different types of operands, we can define functions, whose parameters may involve contextual attributes as well. For instance, given the address (like the street, area code, etc.) of a restaurant, its distance from a current location, measured in a geographic pair of latitude and longitude, can be implemented via function of the form *float Distance(address: string, location: (float, float))* (Thanks to GIS systems, which enable the translation from an implicit reference (such as address) to explicit geographic reference (such as latitude and longitude) [9].) Throughout the paper, we call these privately-defined abstract operators/functions uniformly **contextual operators/functions**.

4 Context-Aware Data Management Strategies

Taking diverse contexts into account, we present the following five context-aware data management strategies.

Strategy 1: *Context as Present On-the-Spot Query Condition.* The highly dynamic, intelligent, and responsive ambient environments prompt users to ask ad hoc queries anytime and anywhere. Such on-the-spot queries usually involve the current context like time, location, traffic, etc. as the query referential points, like "*look for the earliest flight that I can catch*" and "*look for the fastest route to the airport, given the current traffic status.*".

Strategy 2: *Context as Past Recall-Based Query Condition/Target.* For human users, context under which data was accessed in the past is always easier to remember than detailed data content itself. For example, a user might feel difficult to recall the headline of a piece of news. By contrast, the context under which to read the news, such as the place where the news was read, the people present when it was read, or the activity being carried out at the same time, etc., can be easier to remember. In fact, such an observation that context can serve as a powerful cue for recall has a solid foundation in the psychological field, where researchers have developed a theory about *episodic* or *autobiographical memory* [29,18,8].

Strategy 3: *Context as Query Constraint.* From context, we can also infer background knowledge, which can be explored to constrain data requests.
I) *Understanding user's real query intention*. When a user issues a query, s/he usually has some purpose in mind. Database access should be directed by user's specific task. For example, s/he retrieves restaurant information in the city because s/he wants to invite the clients to lunch in a few minutes according to his/her agenda. In this case, only *open* restaurants are meaningful to the user.
II) *Personalizing user's data request*. The usefulness of data is often subjective to the context. For example, for safety reason, a user driving at *mid-night* does not like the database showing him/her the routes, which need to pass through a dark wood. Also, during the daytime *rush hours*, the database system should be considerate enough not to return the routes going through the city center, or the sites that have traffic jams at that moment.
III) *Tuning the level of query content*. The desired abstract level of data to be queried relies on the context as well. For example, a user with a big screen nearby would prefer to display a picture at high-resolution, while with only a tiny handheld PDA, a low-resolution requirement is enough. Also, aggregate/summarized data would be more appropriate than detailed one for small-sized displays.

Apart from assisting query formulation, another important use of context is to determine the manner regarding what, when, where, and how to pass the query output to the user.

Strategy 4: *Context as Criteria for Query Result Measurement.* Given the limitation of small devices, it would be convenient for users if the query answer could be sorted in such a way that the most potentially useful items shown in front. Such a sorting can be performed based on user's interests, obtained from the profile. For example, if the user likes oriental food, the restaurants serving asian meals can be displayed ahead of others.

Strategy 5: *Context as Guide to Query Result Delivery.* The output modality should also be adapted to user's current context. For instance, if the user is driving, it would be convenient to have a speech query output. However, if the user is talking with someone, postponing the delivery of query results by giving a vibration alert, or screen-displaying the query results will be more appropriate. The presentation of query results on a big screen would be welcomed for a group of people who are interested in the query answer.

5 Context-Aware Data Management Infrastructure

We execute the presented context-aware data management strategies via a two-layered architecture, consisting of *public data manager*(s) and a *private data manager*.

5.1 Public Data Manager vs. Private Data Manager

Public data managers can be of any kind of conventional database managers, which support users to access *public databases*, as long as these users have the appropriate rights. In contrast, a *private data manager* behaves like a personal assistant in satisfying user's information needs. It runs on user's private side, say PDA, knows, stores, and manages user's personal information like working area, agenda, profile, data access history, etc. within a *personal/private database*. While conventional public data managers aim essentially at *efficiency* of data management support; the private data manager targets at *usability* aspect by providing the most desirable information to its user in carrying out his/her daily activities. To deliver to the user the right information at the right time in the right way, the private data manager, however, must collaborate with one or more public data managers. As conventional public data managers have been extensively studied in the literature, here, we concentrate on the private data manager, and examine its roles in context-aware data management, particularly context-aware query processing throughout the rest of discussions.

5.2 Major Components of the Private Data Manager

Figure 2 shows the major components of the private data manager. They cooperate as follows.

1) The ***context manager*** defines and maintains the contextual space. It facilitates the use of context by real-time acquiring and instantiating contextual attributes from various external sources.

2) The ***profile manager*** is responsible for managing and supplying user's profile information, including user's interest, working area, habit, preference, etc.

3) The ***agenda manager*** manages user's daily agenda. Given a user's historical database access record, it evokes from the agenda the past associated activities and context for the data access. User's intention in accessing the database can also be predicted based on his/her near-future activities on the agenda.
4) User's database access histories are uniformly stored and managed by the ***log manager***, which collaborates with the agenda manager for recalling the past context and providing support to the context-ware query coordinator.
5) The ***database service communicator*** serves as a bridge between the private data manager and external public data managers. Public data managers must register to it as database service providers in order to be used by the private user. The database service communicator deals with the heterogeneity of different public data managers, while providing to the private user an intermediate uniform view of data model and database schemas in use [1].
6) The ***multi-modal interface*** to external functions/services offers input and output modalities that adaptable to contexts. One such kind of functions is to transform textual output into an audio speech output, when a user is driving.
7) The ***context-aware query coordinator*** enforces and monitors the execution of five context-aware query strategies, presented in Section 4.

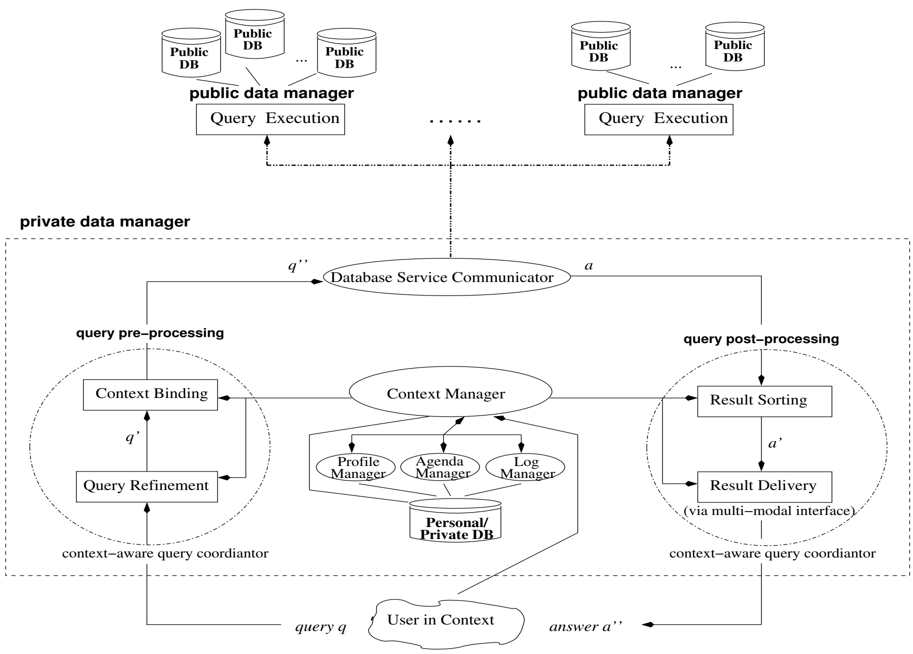

Fig. 2. Steps and components involved in context-aware query processing

[1] In this study, we assume a relational data model is used.

6 Context-Aware Query Processing Framework

A context-aware query is a query whose query answering depends not only on the database, but also on the context under which the query is issued. We can thus view a context-aware query as a parameterized query with two parameters - *database* and *context*, denoted as $CQ(db, [\![\mathcal{C}ontext]\!]) \Longrightarrow A$. The same query, raised by different users, or by the same user under different contexts, may result in different answers. For example, a query on the database relation Route from city "A" to city "B" will return different answers, depending on different contexts. If the query is issued at *mid-night*, the query answer will preclude the route that bypasses an wood for safety reason; however, if the query is issued at *day-time* during the rush hours, the routes that avoid sites having traffic jams *at the moment* will be considered to be useful and thus returned. In this study, we explicate such users' assumptions implicitly behind queries through a set of privately-defined rules, associated with the relations like **Route**. We call these relations **context-sensitive database relations**. Note that, the rules defined here are highly personal, varying from user to user. They are managed and enforced at query time by the context manager at the private data manager's side.

In comparison, a traditional non-context-aware query answering depends only on the database, i.e., $NCQ(db) \Longrightarrow A$.

Given a context-aware query, we divide context-aware query processing into three phases, i.e., *query pre-processing, query execution,* and *query post-processing*, as shown in Figure 2. Except for the *query execution* by the public data manager(s), the rest is performed at the private data manager's side by the *context-aware query coordinator*. The query pre-processing phase proceeds in two steps, i.e., *query refinement* and *context binding*. Before *query execution* can begin, a user's query, which may incorporate context as query condition/target (Strategy 1 & 2), must go through a *query refinement* step whose task is to further constrain the query condition by means of different contextual information (Strategy 3). The *context binding* contacts the context manager to instantiate with exact values all the contextual attributes involved in the refined query, like the current traffic jam, the current time, etc. In this way, the request sent to the public data manager(s) via the database service communicator to execute contains no contextual attributes, and looks like any conventional database query acceptable by the public data manager(s). After the *query execution*, the query post-processing phase will sort the query answer based on user's context via the *result sorting* step (Strategy 4). The *multi-modal interface* component will then invoke external functions/services for the *result delivery* to the user (Strategy 5).

7 Conclusion

In response to *adaptability, dynamicity, personalization,* and *anticipation* demands from AmI, we present five *context-aware* data management strategies. A two-layered infrastructure is outlined to execute the proposed strategies. Different steps and components involved in processing a context-aware query, as well as the query language, are also addressed in the paper. In collaboration with other services like context acquisition component, we are currently building a prototype system which will serve as a user-friendly database frontend.

References

1. A. Asthana, M. Cravatts, and P. Krzyzanowski. An indoor wireless system for personalized shopping assistance. In *Proc. of the IEEE Workshop on Mobile Computing Systems and Applications*, pages 69–74, CA, USA, December 1994.
2. P. Brown, J. Bovey, and X. Chen. Context-aware applications: From the laboratory to the marketplace. *IEEE Personal Communications*, 4(5):58–64, 1997.
3. G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dartmouth College, November 2000. `http://www.cs.dartmouth.edu/~dfk/papers/chen:survey-tr.pdf`.
4. European Commission. Scenarios for ambient intelligence in 2010. `http://www.cordis.lu/ist/istag.htm`, 2001.
5. N. Davies, K. Mitchell, K. Cheverst, and G. Blair. Developing a context sensitive tourist guide. Technical Report Technical Report, Computing Department, Lancaster University, 1998.
6. A. Dey, D. Salber, and G. Abowd. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human Computer Interaction*, 16(2-4):97–166, 2001.
7. M. Dunham and V. Kumar. Location dependent data and its management in mobile databases. In *Proc. of the 9th Intl. Conf. on Database and Expert Systems Applications*, pages 414–419, Vienna, Austria, August 1998.
8. M. Eldridge, P. Barnard, and D. Bekerian. Autobiographical memory and daily schemas at work. *Memory*, 2(1):51–74, March 1994.
9. ESRI. ESRI: GIS and mapping software, 2003. `http://www.esri.com/library/gis/abtgis/giswrk.html`.
10. F. Flynn, D. Pendlebury, C. Jones, M. Eldridge, and M. Lamming. The Satchel system architecture: Mobile access to documents and services. *Mobile Networks and Applications*, 5(4):243–258, 2000.
11. P. Gray and D. Salber. Modeling and using sensed context in the design of interactive applications. In *Proc. of the 8th IFIP Conf. on Engineering for Human-Computer Interaction*, pages 317–336, Toronto, Canada, May 2001.
12. A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. In *Proc. of the 5th ACM/IEEE Intl. Conf. on Mobile Computing and Networking*, pages 59–68, Washington, USA, August 1999.
13. K. Henricksen, J. Indulska, and A. Rakotonirainy. Modeling context information in pervasive computing systems. In *Proc. of the 1st Intl. Conference on Pervasive*, pages 167–180, Zurich, Switzerland, August 2002.
14. J. Hong and J. Landay. An infrastructure approach to context-aware computing. *Human Computer Interaction*, 16(2-4):287–303, 2001.
15. R. Hull, P. Neaves, and J. Bedford-Roberts. Towards situated computing. In *Proc. of the 1st Intl. Symposium on Wearable Computers*, pages 146–153, Boston, USA, October 1997.

16. T. Imielinski and B. Badrinath. Querying in highly mobile distributed environments. In *Proc. of the 18th Intl. Conf. on Very Large Data Bases*, pages 41–52, Vancouver, Canada, 1992.
17. I. Lazaridis, K. Porkaew, and S. Mehrotra. Dynamic queries over mobile objects. In *Proc. of the 8th Intl. Conf. on Extending Database Technology*, pages 269–286, Prague, Czech Republic, March 2002.
18. Barsalou L.W. The content and organization of autobiographical memories. In Remembering reconsidered: Ecological and Traditional Approaches to the Study of Memory (U. Neisser and E. Winograd eds.), Cambridge University Press, 1988.
19. N. Marmasse and C. Schmandt. Location-aware information delivery with ComMotion. In *Proc. of the 2nd Intl. Symposium on Handheld and Ubiquitous Computing*, pages 157–171, UK, September 2000.
20. K. Mitchell. A survey of context-awareness. Technical Report, Lancaster University, UK, March 2002, `http://www.comp.lancs.ac.uk/~km/papers/ContextAwarenessSurvey.pdf`.
21. J. Riekki, J. Huhtinen, P. Ala-Siuru, P. Alahuhta, J. Kaartinen, and J. Röning. Genie of the net: Context-aware information management, 2000. `http://www.pbol.org/projects/genie/~publications/infomanageme.pdf`.
22. D. Salber, A. Dey, and G. Abowd. The context toolkit: Aiding the development of context-enabled applications. Technical Report Technical Report GIT-GVU-98-33, USA, 1998.
23. B. Schilit, M. Theimer, and B. Welch. Customizing mobile applications. In *Proc. of USENIX Mobile & location-Indepedent Computing Symposium*, pages 129–138, Cambridge, Massachusetts, August 1993.
24. W.N. Schilit. System architecture for context-aware mobile computing. PhD thesis, Columbia University, USA, MAY 1995.
25. A. Schmidt, K. Aidoo, A. Takaluoma, U. Tuomela, K. Laerhoven, and W. Van de Velde. Advanced interaction in context. In *Proc. of the 1st Intl. Symposium on Handheld and Ubiquitous Computing*, pages 89–101, Karlsruhe, Germany, September 1998.
26. A. Seydim, M. Dunham, and V. Kumar. An architecture for location dependent query processing. In *Proc. of the 12th Intl. Conf. on Database and Expert Systems Applications*, pages 549–555, Muich, Germany, September 2001.
27. A. Sistla, O. Wolfson, S. Chamberlain, and S. Dao. Modeling and querying moving objects. In *Proc. of the 12th IEEE Intl. Conf. on Data Engineering*, pages 422–432, Birmingham, UK, April 1997.
28. Z. Song and N. Roussopoulos. K-nearest neighbor search for moving query point. In *Proc. of the 7th Intl. Symposium on Advances in Spatial and Temporal Databases*, pages 79–96, USA, July 2001.
29. E. Tulving. *Elements of Episodic Memory.* Oxford University Press, 1983.
30. R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10(1):91–102, 1992.
31. R. Want, B. Schilit, N. Adams, R. Gold, K. Petersen, D. Goldberg, J. Ellis, and M. Weiser. An overview of the PARCTAB ubiquitous computing experiment. *IEEE Personal Communications*, 2(6):28–43, December 1995.
32. B. Zheng and D. Lee. Processing location-dependent queries in a multi-cell wireless environments. In *Proc. of the 2nd ACM Intl. Workshop on Data Engineering for Wireless and Mobile Access (MobiDE01)*, pages 54–65, CA, USA, May 2001.